

16
150pts.

ARUN

Enciclopedia Práctica del Spectrum



Nueva Lente/Ingelek



VARIABLES SUSCRITAS



RECORDEMOS que hasta ahora hemos hablado de dos tipos de variables, las numéricas y las de cadena o alfanuméricas, siendo el empleo de un tipo de variable u otro discrecional, en función de las características del dato que deba albergar. Las diferencias entre ambos tipos de variables son considerables, y afectan tanto a la forma como al fondo.

El nombre de una variable numérica es una serie de caracteres, y su contenido cualquier número, natural, entero o real, pudiendo efectuarse con ellos cálculos matemáticos, y establecer comparaciones, el nombre de una variable de cadena está formado por un solo carácter seguido del símbolo dólar (\$), pudiendo ser su contenido cualquier serie de caracteres, numéricos o alfabéticos, y permitiendo operaciones de unión y traspaso, además de comparaciones.

Por la misma forma en que definimos las variables (diferente nombre identificador), cada una de ellas es independiente de las otras, pudiéndonos considerar como conjuntos aislados de un único elemento. Sin embargo, se plantea el problema de definir grupos de variables que nos permitan una doble posibilidad de tratamiento: una individual y otra de «conjunto».

Normalmente, esta situación se produce cuando

manejamos un cierto volumen de datos relacionados entre sí. Supongamos, por ejemplo, que queremos averiguar la media aritmética de las notas obtenidas por un alumno, a lo largo de los 10 meses de curso, imprimiendo al final un informe con cada una de las notas y la media del curso.

Para ello, debemos almacenar primero las calificaciones en variables por medio de la sentencia **INPUT**. Podríamos hacerlo leyendo líneas por líneas cada una de las calificaciones, y almacenándolas en variables del tipo **N1**, **N2**, ... **N10**, pero esta estructura no nos permitiría hacer uso del bucle **FOR NEXT**, y nuestro programa requeriría 10 líneas de **INPUT** y una considerable pérdida de tiempo en la codificación.

```
10 INPUT "NOTA 1";N1
20 INPUT "NOTA 2";N2
...
100 INPUT "NOTA 10";N10
```

Ya hemos visto algunos de los ventajas que se desprenden del uso de los bucles **FOR NEXT**. Sin embargo, nos hemos dejado quizás lo más importante, la posibilidad de tratamiento de los elementos de un conjunto.



Si se intenta asignar a un elemento de una variable suscrita de cadena, un valor de longitud mayor a la preestablecida, éste se truncará, perdiéndose los *n* primeros caracteres (siendo *n* la longitud máxima) pero no se produce ninguna detención por error.



Las matrices de dos dimensiones se suelen conocer como **TABLAS**.



Si se asigna a un elemento de una variable suscrita de cadena, un valor de longitud menor a la preestablecida, éste es completado con espacios en blanco por la derecha, hasta alcanzar dicha longitud.



Las variables de cadena o numéricas, se pueden considerar como conjuntos aislados de un solo elemento.

! !

CONJUNTOS DE VARIABLES

mos llamar N(1) a la nota del primer mes, N(2) a la del segundo, etc... hasta N(10), para el último de los meses

DEFINICION DE MATRICES

Cuando se efectúa un DIM, desaparece cualquier variable suscrita de igual símbolo que pudiera existir en el momento de la definición, pasando a adquirir la matriz las nuevas orientaciones señaladas, como si nunca hubiera existido la variable suscrita anterior

*

Al definir una variable suscrita numérica, todos sus elementos toman inicialmente el valor cero

*

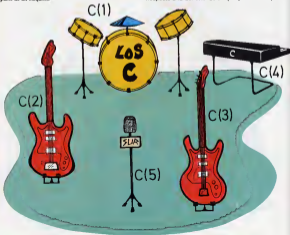
Al definir una variable suscrita de cadenas, todos sus elementos toman el valor inicial de cadenas de espacios en blanco de la longitud preestablecida

*

Las variables suscritas, son generalmente conocidas por el nombre de MATRICES

No cabe duda que la solución ideal a nuestros problemas es definir un conjunto de variables que se denominó «matriz». Esta matriz va a ser de tipo numérico y su nombre genérico será N(), refiriéndose a cada elemento en particular a través de un «índice», que colocaremos entre paréntesis a continuación del nombre de la matriz. A modo general, podemos decir que una matriz es un conjunto de elementos de un mismo tipo, a los que nos referimos por medio del nombre del conjunto, seguido de un índice que identifica a cada elemento en particular. Este índice designa a un elemento determinado por su posición relativa frente a los demás, de forma que todos ellos guardan dentro del conjunto un orden determinado que impone el programador. En el caso concreto de nuestro ejemplo, podría

El BASIC permite tratar definiéndose variables como parte de un conjunto



Para que el BASIC pueda manejar una matriz, es necesario que la hayamos definido previamente. Esta definición consiste en determinar, a través de la sentencia BASIC DIM, el tipo y número de elementos de que se compone, así como su estructura interna. Se trata pues, de una definición algo más compleja que la que debemos hacer cuando nos referimos por primera vez a un nombre de variable, dentro de un programa, a través de la sentencia LET.

La citada operación de definición se conoce como DIMENSIONADO, y produce en el ordenador una reserva del espacio suficiente de memoria para contener la variable, de ahí que en algunas ocasiones, la ejecución de estas sentencias conduzca a un error del tipo 6 Out of memory.

Respecto a la denominación que pueden adoptar

1	OCTUBRE
2	NOVIEMBRE
3	DICIEMBRE
4	ENERO
5	FEBRERO
6	MARZO
7	ABRIL
8	MAYO
9	JUNIO
10	JULIO

N()

estos conjuntos de variables, son evidentemente diferentes según tengan un contenido numérico o de cadena. En el primer caso, el nombre estará formado por una sola letra, que siempre irá seguida de los índices de referencia del elemento, encerrados entre paréntesis:

Tratándose de conjuntos de cadena, habremos de añadir a la norma anterior el consabido símbolo dólar (\$) antes del índice entre paréntesis. En lo referente a este tipo de variables suscritas, existe una restricción muy importante: nunca podemos tener en un mismo momento una variable de cadena y de conjunto de cadenas, que se llamen igual. La razón está bien clara, como ya sabemos, nuestro Spectrum, y en general cualquier artefacto informático, aborrece las ambigüedades, y puesto que el troceado de cadenas se indica encerrando sus parámetros entre paréntesis, podría darse la confusión entre la referencia a un troceado de cadena, o a un índice de variable suscrita. Veamos un ejemplo.

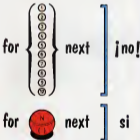
Si suponemos la existencia de una cadena A\$, y una matriz A\$(), puede que queramos conocer el carácter tercero de la cadena A\$, para lo cual ejecutaríamos PRINT A\$(3); sin embargo, idéntica instrucción emplearíamos para conocer el elemento tercero del conjunto A\$() (como variable suscrita). Así pues, el ordenador no puede permitir tal confusión en sus electrónicas «neurona», y se ve obligado a prohibirnos expresamente la utilización de variables de cadena y suscritas que tengan un mismo nombre.

Por otra parte, dado que como denominación de matrices, ya sean numéricas o alfanuméricas, se puede emplear una sola letra, y puesto que, como es habitual en las variables, no existe diferencia alguna entre mayúsculas o minúsculas, el número máximo de variables suscritas de cada tipo es 26 (de la A a la Z), aunque esto no debe resultar un inconveniente insalvable en ningún caso, dado que la posibilidad de manejar múltiples dimensiones con cada matriz, garantiza prácticamente que nunca echaremos en falta variables de este tipo.

En el programa de ejemplo, el conjunto N() aparece las *calificaciones medias obtenidas por un alumno durante los diez meses del curso.*

Las matrices alfanuméricas o de cadena, aborrecen usualmente una dimensión. Su definición consiste en indicar, a través de la sentencia BASIC DIM, su nombre y, entre paréntesis, su número de elementos y longitud del mayor de ellos, separados por comas. A este tipo de matrices unidimensionales, ya sean numéricas o alfanuméricas, se las denominan también VECTORES o LISTAS, puesto que su desarrollo es lineal. Un vivo ejemplo de matriz unidimensional o vector, es el presupuesto anteriormente para la obtención de la nota media en una asignatura. La matriz N() es unidimensional porque, para localizar a un elemento determinado dentro de ella, precisamos de un solo índice, en nuestro caso, la posición relativa de la nota buscada frente a la demás.

Las matrices numéricas admiten más de una dimensión, pero sin embargo no es necesario referenciar la longitud de sus elementos, puesto que



El tratamiento por FOR-NEXT de variables diferentes, no es posible, sin embargo, al es válido el de usar varias variables suscritas.

Un conjunto de variables con algún denominador común, pueden nombrarse en una sola variable suscrita.



!

Las memos de una sola dimensión se denominan comúnmente LISTAS o VECTORES.

*

Las variables suscritas son conjuntos de variables con un mismo nombre genérico, distinguidas por el ordenador mediante un número entre paréntesis, denominando índice.

*

Las variables suscritas pueden ser numéricas o alfanuméricas.

*

La denominación general de una variable suscrita numérica, debe ser una sola letra, tanto en el caso de las variables individuales, no sea distinguidas entre mayúsculas y minúsculas, por lo cual el máximo número de variables suscritas diferentes es 26 (de la A a la Z).

todos los variables numéricas BASIC ocupan lo mismo, con independencia de su contenido. Esto que en un principio nos pueda parecer algo difícil de comprender, no nos debe preocupar ahora es simplemente un hecho, que bastante más adelante estudiaremos.

De esta forma, una matriz numérica queda definida por una sentencia DIM, al indicar su nombre y, a continuación y entre paréntesis, su número de elementos y dimensión, separados por una coma. A continuación se muestran algunos ejemplos de definición de diversas matrices.

10 DIM A\$(10,20), DIM A(200), DIM B(200,10)

En este línea de programa se han dimensionado tres matrices. La primera de ellas, A\$, es de cadena y consta de 10 elementos de longitud máxima 20. La segunda, A, es numérica y consta de 200 elementos, en una dimensión, es decir, es

La sentencia DIM se utiliza para reservar espacio al declarar a una variable variable.



un vector. Por último, B, es otra matriz numérica de 200 elementos, cada uno de ellos desdoblado en otros 10, siendo por lo tanto tridimensional. Para tener un concepto más claro de lo que implica la multidimensionalidad de una matriz, podemos pensar en una ampliación del problema propuesto anteriormente. Supongamos que, en vez de calcular la nota media de las calificaciones de un alumno, deseamos obtener además las de toda la clase.

En este caso, es evidente que podríamos aprovechar gran parte del programa anterior. De hecho, nuestro problema es el mismo que el propuesto al principio, pero multiplicando por el número de alumnos de la clase. Se impone, pues, el uso de una matriz bidimensional $N(i,j)$, para el almacenamiento de esta estructura de datos, que deberemos definir de la forma DIM N(50,10), suponiendo una clase de 50 alumnos y un período a evaluar de 10 calificaciones.

De este modo, N(1,1) representará la primera calificación del primer alumno, N(6,6) la cuarta del



Cada uno de los elementos del conjunto, se distingue de los demás por su índice entre paréntesis.

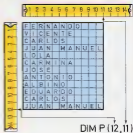
sexto alumno y, por último, N(50,10) la décima del que hace el número cincuenta.

Si ampliamos aún más nuestro ejemplo sobre las notas de los alumnos, haciéndolo extensivo a todas las clases que componen el curso, nos encontramos ante el caso de una matriz tridimensional, que podríamos representar como:

DIM N(8,50,10)

En este caso, sería preciso el empleo de 3 índices para calificar a un elemento determinado, es decir, el número de aula, el número de alumno y de qué calificación se trata.

Las variables azules a continuación, se declaran indicando el número de elementos que las componen y su longitud máxima.



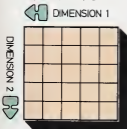
CONTENIDO DE LAS MATRICES

Las matrices, teóricamente, deben definirse una sola vez en el programa. Han de colocarse, por ello, en un lugar concreto del programa por el que sólo se pase una vez. Explicamos el porqué de ese «reforzamiento» del comienzo de párrafo. Cuando una matriz ha sido definida y contiene datos, si se efectúa una nueva definición de la misma matriz a través de otra sentencia DIM, se pierde su contenido, adoptando la variable suscrita la nueva configuración indicada, y su contenido pasa a ser ceros si es numérica, y blancos si es de cadena.

Este efecto que «borramos» no es deseable, puede ser utilizado por nosotros mismos, a nuestra conveniencia, puesto que el «redimensionado» no producirá ninguna detención por error, sino simplemente el efecto descrito. En todo caso, lo habitual es no desear el borrado sobre el contenido de una matriz, debido a lo cual, deberemos tener cuidado con el lugar en que situemos las definiciones.

UN EJEMPLO

Para aclarar ideas, proponemos el siguiente programa de ejemplo, que nos pregunte algunos



En las matrices unidimensionales, todos los elementos ocupan la misma memoria, independientemente de su contenido.

En variables suscritas pueden abarcar más de dos dimensiones, en cuyo caso se las suele conocer como MATRICES DE n DIMENSIONES.



Las matrices bidimensionales, se conocen como TABLAS.

Las variables suscritas unidimensionales, se conocen como LISTAS o VECTORES.



!

miento los nombres de 10 capitales europeas, doblando por cada acierto nuestra puntuación, y dejándola en la mitad a cada fallo, para alcanzar una evaluación final de los resultados:

```

10 REM CAPITALAS EUROPEAS - J.M. LOPEZ MARTINEZ
20 LET X$=""          " LET P=1
30 DIM P$(22,32)
40 FOR I=1 TO 22
50 READ P$(1),P$(1,17 TO )
60 NEXT I
70 FOR I=1 TO 10
80 LET X=INT (RND*22)+1: IF P$(X,32)="
90 LET P$(X,32)="X"
100 INPUT (P$(X, TO 14)) : LINE X#: TO 1
5)
110 IF X$=P$(X,17 TO 31) THEN LET ES="CORRECTO": LET P=P+P/60 TO 130
120 LET ES="LO SEEMO": LET P=P/2
130 PRINT ES, "ACUMULADO": INT P
140 NEXT I
150 IF P>=512 THEN PRINT "ENHORBUENA!"
POR LOS " INT P1 " PUNTOS" GO TO 170
160 PRINT INT P2 " PUNTOS. MAY QUE ESTUS [AR MAS]
170 STOP
180 DATA "ISLANDIA", "REYKJAVIK", "NORUEA", "OSLO", "SUECIA", "ESTOCOLMO", "FINLANDIA", "HELSINKI", "U.R.S.S.", "MOSCÚ"
190 DATA "GEMANARCA", "COPENHAGUE", "POLONIA", "VARSOVIA", "R.D.A.", "BERLIN", "R.F.A.", "BONN", "AUSTRIA", "VIENA"
200 DATA "CHECOSLOVACIA", "PRAGA", "IRLANDIA", "DUBLIN", "REINO UNIDO", "LONDRES", "FRANCIA", "PARIS", "SUIZA", "BERNA"
210 DATA "HUNGRIA", "BUDAPEST", "RUMANIA", "BUENAREST", "YUGOSLAVIA", "BELGRADO", "PORUGAL", "LISBOA", "ESPAÑA", "MADRID"
220 DATA "ITALIA", "ROMA", "GRECIA", "ATENAS"
    
```



El número de DIMs en un programa, así como su ubicación en el mismo, no tienen limitación alguna.

En el programa CAPITALES DE EUROPA, tener ubicado el símbolo abreviado (\$) como comento que está la repetición de programas

*

Para evitar confusiones entre índices y/o números de arrays (trascodo de columnas), es no posible hacer coexistir en la memoria una variable sujeta de cada una y una columna individual, con el mismo nombre.

*

La definición inicial de las variables sujeta, se lleva a cabo mediante la sentencia DIM, en la cual se indican las características de la misma nombre, y por tanto, sus (número o abstracción), longitud y número de dimensiones.

Este programa tiene de todo. Entre otras cosas, algunos conceptos nuevos sobre temas que ya hemos comentado y aprovechamos ahora para ampliar:

Los conceptos de variables abstracción, se describen mediante una sola línea seguida el símbolo dólar (\$) :



Para la primera novedad, debemos fijarnos en las líneas 20 y 100. En la línea 20, asignamos el valor de 15 espacios a la variable X\$. Esto es debido a que X\$ es la variable que empleamos para el INPUT de la capital en la línea 100, de la forma X\$ (TO 15). Generalizando, podemos conseguir variables de longitud fija utilizando este procedimiento: primero definir la variable con tantos blancos como deseemos, 15 en nuestro caso, y después hacer las posteriores asignaciones de valor por medio de LET, READ o INPUT, especificando qué parte de X\$, en nuestro ejemplo TO 15.

Este modo de trabajar nos es muy útil en esta ocasión, puesto que, como sabemos, los elementos de las tablas alfanuméricas son todos de la misma longitud, es decir, se rellenan con blancos las posiciones no utilizadas. En este estado de cosas, la única manera de establecer comparaciones es este sistema. Para completar nuestro conocimiento sobre el comportamiento de los dimensionados alfanuméricos, hemos de saber que, al igual que las asignaciones de menor longitud se rellenan con blancos por la derecha, los más largas se truncan por la longitud máxima admitida.

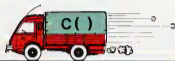
Así por ejemplo, si a un vector alfabético al que se le señala una longitud máxima por elemento

de cinco caracteres, se le intenta asignar el valor "ANA", pasará a tener el contenido "ANA ". Por el contrario, si se le asigna "GLORIA", tomará el valor "GLORI".

En la línea 30, definimos el vector alfanumérico $P(22,32)$, es decir, 22 elementos de 32 posiciones.

En las líneas 40 a 60, se establece la carga de la matriz, a partir de los datos contenidos en las líneas de data, 180 a 220. Como vemos, cada elemento de la matriz está dividido en dos mitades, de las cuales la primera (posiciones 1 a 16) contiene el país y la segunda (posiciones 17 a 32) contiene la capital.

En la línea 70 comienza el bucle de diez preguntas. Es evidente que precisamos una elección de preguntas al azar, pero sin que éstas puedan



Una matriz numérica debe tener su nombre compuesto por uno solo letra.

laución de la respuesta, y el acumulado de puntos obtenidos hasta el momento, y en la línea 140 se cierra el bucle de preguntas abierto en la instrucción 70. Por último, las líneas 150 a 160, que sólo se alcanzan al final del ciclo de preguntas, evalúan los resultados obtenidos en base a la puntuación acumulada.

El programa en su estado actual, memoriza hasta 22 países europeos con sus respectivas capitales. Para ampliar sus conocimientos geográficos, sólo se necesita añadir las DATA correspondientes, teniendo en cuenta que también hemos de aumentar el número de elementos en el dimensionado de la matriz, y por supuesto, el bucle de lectura de las DATA, para su asignación a la variable susunto. Del mismo modo, la extracción aleatoria de la pregunta también debe ser alterada, puesto que se mueve en el rango antes citado.



Cuando se us INPUT, la variable aparece entre paréntesis, en consideración de salida (output).

repetirse. Esto se logra con las instrucciones 80 y 90. En 80, seleccionamos primero un número aleatorio entre 1 y 22 y comprobamos si tiene la marca de haber aparecido ya. Si la tiene, repetimos la generación aleatoria, si no, pasamos a la instrucción 90, donde marcamos el elemento de la tabla como utilizado, con un símbolo almohada (^) en la posición 32.

En la línea 100 utilizamos una nueva versión de INPUT. Es la modalidad de petición con texto variable, que consiste en indicar a continuación del INPUT, y entre paréntesis, el nombre de la variable que deseamos aparezca en la petición de los datos. Es muy importante que esta variable aparezca entre paréntesis, para distinguirla de la que se va a asignar en el INPUT.

En la línea 110, comparamos la respuesta tecleada (de longitud fija), con las posiciones 17 a 31 del elemento (X) seleccionado. En caso de acierto, se asigna a E\$ su valor, y se duplica el total acumulado de puntos, desviando la ejecución a la instrucción 130. A la línea 120 solo se accede en caso de error, para colocar E\$ en su valor, y dejar el acumulado de puntos en la mitad. En la línea 130 se imprime el valor de E\$, evi-



Cuando una variable numérica se dimensiona, sus elementos adoptan automáticamente el valor cero.

Cuando una variable sujeta de cadena se dimensiona, sus elementos toman el valor inicial de cadenas de espacios en blanco.



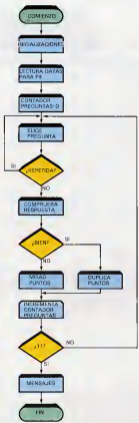


!!

En las variables asociadas a los números, el paréntesis que corresponde a la segunda dimensión en la sentencia DIM, se emplea para expresar la longitud máxima de los elementos de la variable.

*

Al contrario que las cadenas individuales, los elementos de una variable sujeta a una longitud fija, definida en su declaración DIM



Organización del programa CAPITALES DE EUROPA.



La segunda variable sujeta a depender completamente el valor de la misma variable.

Es una variable de cadena y una matriz del mismo tipo, así como coincide con su declaración.



Finalmente, para que la filosofía general del programa se conserve, hemos de respetar la distribución de los datos en el vector, es decir, la primera mitad de cada línea ha de contener el país, y la segunda, la respuesta.

Como demostración de la fiabilidad que es modificar este programa, añadiremos cinco preguntas más a nuestro test geográfico: las capitales de BULGARIA (SOFIA), ALBANIA (TIRANA), BELGICA (BRUSELAS), PAISES BAJOS (LA HAYA) y LUXEMBURGO (LUXEMBURGO). Siguiendo los pasos que hemos mencionado anteriormente, deberemos completar las líneas DATA que finalizan el programa con los nuevos datos:

```

330 DATA "BULGARIA", "SOFIA", "ALBANIA", "TIRANA", "PAISES BAJOS", "LA HAYA", "BELGICA", "BRUSELAS", "LUXEMBURGO", "LUXEMBURGO"
  
```

Además, es necesario adaptar las dimensiones de la matriz, para que pueda albergar estos nuevos elementos, y por supuesto, prolongar el bucle de lecturas (READ), puesto que el simple hecho de añadirlos a la DATA no implica que sean leídos.

```

30 DIM P$(27,32)
40 FOR I=1 TO 27
  
```

Por último, es preciso ampliar el rango de selección aleatoria de las preguntas, para que también sea posible escoger las recién llegadas:

```

80 LET X=INT (RND*27)+1:IF P$(X,32)="" THEN GO TO B0
  
```



¡SE RUEDA!



En este capítulo daremos ya por cerrado el tema del dinamismo en la pantalla completa, con una técnica basada en la subrutina de transformación de memoria que vimos anteriormente. Lamentablemente, debido a la gran cantidad de memoria que se precisa, la técnica que a continuación exponemos está vedada para los usuarios del modelo de 16 K.

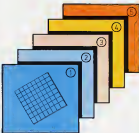
Realmente, su fundamento es muy sencillo. Se trata de almacenar varias pantallas en la memoria, por supuesto no en la memoria de pantalla, para posteriormente situarlas en esta zona una después de otra, de forma que consigamos una animación similar a la que se obtiene en el cine con el paso de los fotogramas. La restricción en nuestro caso está bien clara: el máximo número de pantallas que se puedan almacenar son cinco, y con sólo cinco fotogramas mal vamos a hacer un largometraje.



El sistema de extracción que presentamos en pantalla en este capítulo, es similar al utilizado para la proyección de películas de cine.

No obstante, el efecto que se consigue con esta técnica es muy digno de tener en cuenta. El primer paso será obtener los necesarios cinco fotogramas, a tal fin, podemos introducir el programa de gráficos en tres dimensiones, aparejado en

el capítulo segundo de la sección de programas. Una vez hecho esto, iremos creando pantallas de acuerdo con las fórmulas que a continuación se facilitan, y por el mismo orden. Cada pantalla deberá ser grabada una detrás de otra en una misma cinta, para que luego puedan ser utilizadas por el programa de demostración "CINEMA".



Contra a la subrutina de transformación de código máquina, la construcción es sencilla por superposición de imágenes a alta velocidad.

La grabación de las imágenes es muy fácil. Sólo hemos de copiar tranquilamente a que la figura correspondiente se dibuje, y cuando aparezca el mensaje **9 STOP statement, 470-1** podremos grabar la pantalla con el comando **SAVE "Nombre">"SCREEN#**. Lógicamente, el nombre para la grabación no tiene la menor importancia, y puede ser por ejemplo, el número de "fotograma" ("1", "2", "3", etc.).

Para que aparezca el mensaje de interrupción del programa, que nos dará opción a la grabación de la pantalla, es necesario, previamente a la ejecución del programa **30**, introducir una pequeña modificación: cambiar la línea **470** para que quede como **470 STOP**.

Bien, ahora hemos de hacer el reparto de pago los para nuestra película: la función para la primera pantalla será $(X^*X+Z^*Z)/1000$, para la segunda pantalla utilizaremos $(X^*X+Z^*Z)/3000$,

BITS

Habitualmente, el color que el Spectrum utiliza como fondo para el marco de la pantalla, es el de contraste, si queremos utilizar otro color debemos de indicarlo mediante un código POKÉ a la Variable del Sistema **BORDER (23624)**. El valor a introducir será el código de color del marco multiplicado por ocho, más el código de área a emplear. Así por ejemplo, **POKÉ 23624,52** producirá la impresión en tinta verde sobre marcos amarillos, puesto que $52 = 8 * 6$ (código amarillo) + 4 (todas verde). Este efecto desaparecerá al ejecutar un código POKÉ a esta dirección, o al ejecutar cualquier comando **BORDER**.

BITS

La lectura de información del disco no está sólo reservada al código máquina. Nosotros mismos, desde el BASIC, podemos organizar mediante una sencilla operación, si el Spectrum está "en funcionamiento" algún sonido. Tal y como se hace en el siguiente ejemplo, esta técnica puede emplearse para emular un programa hasta la carga del programa, con la seguridad de que, sea cual sea el espacio entre ambos, no perderemos ni un bit de información.

```

10 BORDER 1
PAPER 1 CLS
30 FOR I=0 TO 6
30 BEEP 01.30
BND*2% BORDER
6-1
40 LET J=IN 254
IF J>127 THEN LET
J=J-128
60 IF J>63 THEN
LOAD ""
80 NEXT I GO TO
20
    
```

para la tercera O (dibujo plano), para el cuarto "fonograma" SIN ((60-X*X-Z*Z) * (ABS SIN (60-X*X-Z*Z)-1))/5000), y para el último, SIN (((60-X*X-Z*Z)*(ABS SIN (60-X*X-Z*Z)+1))/2000)

Una vez preparada la película, ya sólo queda proyectarla, para lo cual emplearemos el programa CINEMA

```

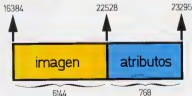
10 FOR CINEMA - C:DE LA OSSA A F. LOPEZ
MARTINEZ
20 GVAR 24664
30 LOAD "S"CODE 34667
40 LOAD "2"CODE 48791
50 LOAD "3"CODE 46925
60 LOAD "4"CODE 50879
70 LOAD "5"CODE 59232
80 FOR I=23296 TO 23587
90 READ A
100 POKE I,A
110 NEXT I
120 DATA 99,9,9,17,9,64,1,9,24,237,176,
241
130 LET I=1
140 LET A=2
150 LET S=34667+A*6145
160 POKE 23296,INT (S/256)
170 POKE 23297,S-256*PEEK 23296
180 LET S=USR 23296
190 BEEP .1,-28160*60
200 LET A=A+1
210 IF A=8 OR A=9 THEN LET I=I+1: LET
A=A+2: I
220 GO TO 150
    
```

Lo primero que hace el programa al ejecutarse, es cargar las cinco imágenes que configuran la película, para acto seguido, y cíclicamente, ir las proyectando en la pantalla, generando la animación de una cama elástica, o al menos "algo por el estilo".

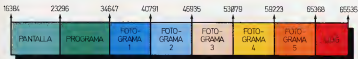
Obviamente, el programa CINEMA funcionará idénticamente igual, cualquiera que sean los fotogramas que se estén en la memoria, por tanto, podemos convenirnos en mal-George Lucas, y protagonizar la dirección de apasionantes secuencias a nuestro gusto: el estallido de una nave espacial, el movimiento de un brazo, el pestañeo de unos ojos, etc... En nuestro caso hemos empleado el programa 30, pues con un corto esfuerzo de introducción, nos permite disponer de las cinco fotografías necesarias. Indudablemente,



Para hacer nuestra película, disponemos con sólo de cinco fotogramas



Para hacer nuestra película, sólo se han almacenado los archivos de imagen, pero no los datos de atributos; por tanto, nuestras películas son en blanco y negro.



En el gráfico se ve la distribución de memoria en el programa CINEMA.

ta, otras secuencias considerablemente más arduas pudieron ser obtenidas, sin embargo, su espectacularidad no justificaba la gran ocupación de espacio, y lo absurdo de su introducción. Como hemos visto, el programa CINEMA dejó un camino abierto a una actividad totalmente creativa y artística, gracias a un ordenador. El punto más importante en este aspecto, es que los fotogramas cabían lo suficientemente próximos en el tiempo, como para que no se produjeran saltos en la imagen, sino una armoniosa sensación de continuidad.

Sobre la subrutina de transferencia, sólo hay que añadir que es completamente reutilizable, debido a lo cual, funcionará perfectamente cualquiera que sea la zona de memoria en que se almacene. Para ahorrar espacio, en el ejemplo ha sido depositada en el *buffer* (memoria intermedia) de la impresora. Por otra parte, el sistema de utilización de la ratina es muy simple: el primer código es siempre un 33, al cual le sigue la dirección en que se encuentra la pantalla a transferir, expresada en peso bajo peso alto.

Los próximos tres códigos, no se cambian (17, 0 y 64), puesto que indican la dirección de destino de la transferencia, lo cual, lógicamente, es siempre la misma: la pantalla (16384). Por último, al próximo código (1), le debe seguir, expresado nuevamente en bajo-alto, el número de bytes a transferir, en nuestro caso son 8144, es decir, el archivo de imagen, y por tanto, los bytes codificados con 0 y 24 (24*256=0-6144). Si quisiéramos transferir la pantalla completa, incluyendo

el área de atributos, los bytes a codificar serían 0 y 27, correspondientes a los 6912 bytes que integran la pantalla completa (27*256=0-6912).

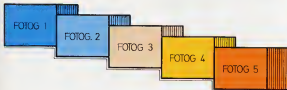
CINEMASCOPE

En la introducción al dinamismo de pantallas completas, comentamos la posibilidad de, utilizando los mismos técnicas, poder animar una determinada zona de la pantalla, de fácil tratamiento por la subrutina de transferencia. Ya entonces, sugerimos para la creación de estos efectos la utilización de textos de pantalla, que debido a la peculiar organización del archivo de imagen, podían ser tratados prácticamente como una pantalla independiente.

Como vimos entonces, para la consecución de esta técnica de dinamismo, no son importantes las dimensiones del área a tratar, sino el hecho de que los imágenes sean transferidas completas, es decir, sin tomar en consideración qué fi

Por motivos de memoria, la técnica descrita en este capítulo, está limitada a los usuarios del modelo de 16 K.

Debido a las direcciones de almacenamiento, aunque las pantallas se cargan completas, sus respectivos archivos de imagen ocupan el área de atributos de la pantalla.



SIMON



ESTE nuevo programa nos ayudará a ejercitar nuestra capacidad de retención, tanto visual como auditiva, al mismo tiempo que nos divertiremos y pasaremos un rato muy agradable.

El juego del SIMON es una competencia contra nosotros mismos, es decir, un solitario, en el cual la máquina tiene un papel pasivo, de forma similar al MASTER MIND (MASTER CLUB, para los lectores de esta Obita).

El ordenador emite un determinado sonido, al tiempo que presenta un cuadrado de color en la pantalla, a continuación debemos pulsar la tecla correspondiente a ese color, o si lo preferimos, a ese sonido. Si conseguimos acertarlo, al sonido y color anteriores se añadirá uno nuevo, diferente o no. Esta vez debemos repetir los dos colores. El juego continuará aumentando la secuencia, hasta que nos equivoquemos en alguna de ellas. Nuestro programa posee dos niveles de dificultad. El primero y más sencillo, hace uso de los cuatro primeros colores básicos del Spectrum:

azul, rojo, magenta y verde. El nivel restante, utiliza dos colores más, siguiendo el orden antes expuesto: azul, rojo, magenta, verde, cian y amarillo.

Estos colores son generados aleatoriamente uno a uno, y almacenados en una tabla para su posterior presentación en pantalla. Por cada secuencia de colores acertada, nuestro ordenador elige otro, añadiéndole al final de la cadena de colores, que va aumentando a medida que avanzamos en el juego.

Por cada color generado, se emite siempre un mismo sonido, de forma que si no somos capaces de recordar la secuencia policromada presentada en la pantalla, nuestro Spectrum nos da otra

La presentación del programa en la pantalla, que coincide en totalidad con la utilización de la subrutina de caracteres pagadas.



!

El programa carece de gráficos dibujados, debido a lo cual en el teclado no aparecen caracteres subrayados.

*

Para grabar el programa, utilizaremos la acción combinada de los caracteres SAVE y LINE de la siguiente forma:
SAVE "SIMON"
LINE 1400.

opción: recordarla dependiendo de la secuencia de notas que «deja» escapar. En el momento en que fallemos un color o nota, el Spectrum nos lo hará saber inmediatamente, a la vez que nos informará de los puntos obtenidos, finalizando el juego en ese momento.

La forma de puntuar de nuestro Spectrum es la siguiente: sumaré al marcador tantas puntos como notas o colores tenga la secuencia presentada, siempre y cuando sea cierta dicha secuencia. Es decir, si el ordenador ha generado 7 colores y hemos sido capaces de recordarlos por su orden cronológico de aparición, sumaré 7 al marcador, mientras que si fallamos en alguno de ellos, la puntuación en esa secuencia será 0 y la partida habrá finalizado.

EL PROGRAMA

Este programa puede almacenar una serie de 200 colores con sus notas correspondientes. Si queremos aumentar o disminuir dicha cantidad, no tenemos más que variar el número que define la longitud de la matriz S, que lógicamente está inicializada en 200 elementos: DIM S(200) (línea 130).

El nivel más sencillo de nuestro juego del SIMON, se basa en los cuatro primeros colores del Spectrum.

!

Los dos niveles de dificultad que posee el juego, hacen referencia al número de colores que pueden entrar en la secuencia (cuatro o seis).

*

Los teclados que hay que pulsar para entrar la secuencia de colores y suavidos, son los de los números correspondientes a los edificios de color dicho.



El tiempo que un color esté presente en la pantalla, depende del nivel de dificultad, y se controla mediante la variable T, que se utiliza como parámetro del comando BEEP. Esta variable numérica, según el nivel de dificultad elegido, puede tener dos valores: en el nivel uno, T adquiere el valor 0.5, y en el segundo nivel su valor es 0.05. De esta forma, cuanto más pequeño sea T, menos tiempo estará el color generado en la pantalla.

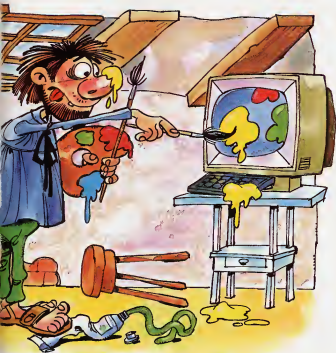


En SIMON volvemos a utilizar la subrutina de caracteres gigantes de PSION, custodiando así la estética del programa en la pantalla. Por tanto, seguiremos para la adopción de la rutina las especificaciones que a este respecto se han breñado en programas anteriores, que también hayan hecho uso de ella.

En esta ocasión, veremos de nuevo los caracteres subrayados en el listado, puesto que no se

emplean gráficos, sino diferentes códigos de PA-
PER para la generación de los colores.

Es importante recordar la absoluta necesidad de tener cargada la subrutina de caracteres gigantes para el correcto funcionamiento del programa, así pues, para almacenarlo, utilizaremos el siguiente comando: **SAVE "SIMON" LINE 1480**. Dependiendo la autoejecución en la línea 1480, que es la encargada de la lectura del código máquina.



```

10 REM *****
20 REM = J.P. RAYONAL SERRANO =
30 REM *****
40 REM = Programa. GORIN =
50 REM *****
60 REM INICI. - VARIABLES
70 BORDER 0: PAPER 0: INK 9: CLS : FORD 20650,9
80 LET X0=100
90 LET Y0=0
100 LET C=10
110 GOSUB 101
120 LET X0=0
130 GOSUB 100
140 GOSUB 100
150 REM MENÚ DE PRESELECCION
160 PRINT AT 10,3: FLASH 1: INK 6: PAPER 2:1
170 PRINT AT 10,7: "SIN COMPLICACIONES"
180 PRINT AT 12,2: FLASH 1: INK 2: PAPER 6:2
190 PRINT AT 12,7: "CON BASTANTES PROBLEMAS"
200 PRINT AT 5,7: PAPER 5: INK 1: "NIVEL DE DIFICULTA
210 PRINT AT 21,1: " * * * * * ELIGE OPCION * * * * "
220 LET X0=INKEY$
230 IF CODE X0=50 OR CODE X0=69 THEN GOSUB 200: GO
240 BEEP ,2,80
250 GO TO 220
260 REM AMPLIAR LINEA 230
270 IF C=23 THEN LET C=22
280 PRINT INK 4: OVER 1:AT 21,C: "
290 BEEP ,0,48
300 PRINT INK 7: OVER 1:AT 21,C-1: "
310 BEEP ,0,45
320 LET C=C+1
330 RETURN
340 REM NUM. DE COLORES
350 LET CLAS=2+(VAL X0+1)
360 IF CLAS=4 THEN LET T=5: GO TO 360
370 LET T=65
380 CLS
390 REM NUM. GONI. COMENZ0
400 PRINT AT 19,8: "CON CUANTOS TONOS DEBES EMPEZAR"
410 PRINT AT 21,12: "1 - 5"
420 LET X0=INKEY$
430 IF CODE X0=69 OR CODE X0=50 THEN GO TO 420
440 BEEP ,2,60: CLS
450 LET D0=VAL X0
460 PRINT AT 0,5: INK 3:AS
470 REM DIBUJO NARDEN
480 IF CLAS=6 THEN GO TO 590
490 FOR N=3 TO 13 STEP 5
500 PRINT PAPER 7:AT 0,N: "
510 NEXT N
520 PRINT AT 0,5: INK 3:AS
530 FOR N=3 TO 13
540 FOR N=3 TO 13 STEP 5
550 PRINT PAPER 7:AT N,N: "
560 NEXT N
570 NEXT N
580 GO TO 700
590 FOR N=3 TO 13 STEP 5
600 PRINT AT 0,5: INK 3:AS
610 PRINT PAPER 7:AT N,N: "
620 NEXT N
630 FOR N=3 TO 13
640 FOR N=3 TO 13 STEP 5
650 PRINT PAPER 7:AT N,N: "
660 NEXT N
670 PRINT AT 0,5: INK 3:AS
680 NEXT N
690 DATA "1","2","7","1","2","6","7","2","5","0","2","4","N","2","3
700 RESTORE 690
710 FOR O=1 TO 5
720 LET X0=O+10
730 LET Y0=Y0+10
740 READ P0,N0,Y0:
750 BRIGHT 1: INK P0-1
760 GO SUB 1000
770 NEXT O
780 INK 9: BRIGHT 0
790 REM PROGRAMA PRINCIPAL
800 FOR N=1 TO CNT-1
810 LET S=INT(32*INT (RND*(CLS)+1
820 LET S0=INT(0
830 NEXT N
840 REM CONTIENE SONIDOS
850 PRINT AT 0,5: INK 3:AS
860 LET S=INT(0
870 FOR N=1 TO 300
880 LET K=STR$ S%1: GO SUB 1150
890 BEEP T,S%1+7: GO SUB 1200
900 NEXT N
910 LET S0=INT(0
920 REM EVENTOS ADVERSARIOS
930 FOR N=1 TO 300-1
940 LET K=INKEY$
950 IF CODE K=69 OR CODE K=5000 THEN CLRS THEN GO
960 TO 940
970 IF VAL K=0+1 THEN GO SUB 1150: BEEP ,6,S%1+7
980 SUB 1200: GO TO 900
990 GO TO 1070
1000 NEXT N
1010 LET A=INT(0
1020 PRINT AT 10,10: "MUY BUEN"
1030 PRINT AT 21,10: "CONTINUO"
1040 FOR P=1 TO 7: BEEP ,1,50: NEXT P
1050 PRINT AT 10,10: "
1060 PRINT AT 21,10: "
1070 GO TO 840
1080 REM FALLOS
1090 GO SUB 1150: BEEP 1,-10: LET P0="M011": LET Y0=
1100 LET X0=0: LET Y0=1: INK 6: GO SUB 1200
1110 LET P0="OTRA PARTIQUITA ": LET Y0=175: LET X0=0
1120 LET Y0=1: INK 5: GO SUB 1200
1130 IF INKEY$="N" THEN RETURN
1140 IF INKEY$="Y" THEN GO TO 1120
1150 GO TO 1000
1160 REM PUNTO DE CURSOS
1170 LET O=VAL X0
1180 IF O=1 THEN LET P=0: LET C=0: GO TO 1240
1190 IF O=2 THEN LET P=0: LET C=0: GO TO 1240
1200 IF O=3 THEN LET P=0: LET C=10: GO TO 1240
1210 IF O=4 THEN LET P=0: LET C=0: GO TO 1240
1220 IF O=5 THEN LET P=0: LET C=10: GO TO 1240
1230 PRINT "ERROR": STOP
1240 FOR P=0 TO P+3
1250 PRINT PAPER 0:AT P,C: "
1260 NEXT P
1270 RETURN
1280 REM BARRIDO
1290 LET D=0: GO TO 1240
1300 REM : ROM : ROM
1310 RESTORE 1300
1320 DATA 70,44,32,77,46,32,77,65,09,77,62,65,74,32,0
1330 FOR N=1 TO 21
1340 READ A: LET A=N+O*80+R A
1350 NEXT N
1360 RETURN
1370 REM BASIC CHARACTER, SIGANTES
1380 LET X0=1256-O*80+LEN A*5/2
1390 LET Y0=23200
1400 POKE 1+X0:POKE 1+Y0:POKE 1+X0:POKE 1+Y0
1410 LET Y0=40: LET W=LEN A
1420 FOR O=1 TO W
1430 POKE 1+X0:POKE 1+Y0
1440 NEXT O
1450 POKE 1+X0+1:200
1460 LET W=USR 32256
1470 RETURN
1480 REM AUTO-LEC. OBLIGATORIA
1490 CLEAR 32255
1500 LOAD "LET"CODE
1510 RUN

```